学校编码：10384　　分类号__密级__
学号：X2007222005　　UDC___

厦门大学

工 程 硕 士 学 位 论 文

J2EE 环境下 WEB 应用的代码安全问题研究及实践

Code Security Research and Practice for Web Application under J2EE Environment

洪昕

指导教师姓名：周剑扬副教授
林龙富高级工程师
专　业　名　称：电子及通信工程

论文提交日期：
论文答辩时间：
学位授予日期：

答辩委员会主席：__
评阅人：__

年　　月

厦门大学学位论文原创性声明

本人呈交的学位论文是本人在导师指导下，独立完成的研究成果。本人在论文写作中参考其他个人或集体已经发表的研究成果，均在文中以适当方式明确标明，并符合法律规范和《厦门大学研究生学术活动规范（试行）》。

声明人（签名）：

年　月　日

厦门大学学位论文著作权使用声明

本人同意厦门大学根据《中华人民共和国学位条例暂行实施办法》等规定保留和使用此学位论文，并向主管部门或其指定机构送交学位论文（包括纸质版和电子版），允许学位论文进入厦门大学图书馆及其数据库被查阅、借阅。本人同意厦门大学将学位论文加入全国博士、硕士学位论文共建单位数据库进行检索，将学位论文的标题和摘要汇编出版，采用影印、缩印或者其它方式合理复制学位论文。

本学位论文属于：

（　　）1.经厦门大学保密委员会审查核定的保密学位论文，于　　年　　月　　日解密，解密后适用上述授权。

（ √ ）2.不保密，适用上述授权。

（请在以上相应括号内打“√”或填上相应内容。保密学位论文应是已经厦门大学保密委员会审定过的学位论文，未经厦门大学保密委员会审定的学位论文均为公开学位论文。此声明栏不填写的，默认为公开学位论文，均适用上述授权。）

声明人（签名）：

年　　月　　日

目录

Table of Contents

摘要

随着互联网的不断普及与发展，越来越多的应用从传统模式延伸至网络。人们可以方便的通过网络进行银行转账，产品购买。这其中大部分的应用是基于Web 浏览器的,Web 应用程序的增多使得其安全问题变得越来越受关注。由于 Web 应用采用的是典型的客户端/服务器模型，客户端向服务器发出请求，服务器通过执行操作（如将信息发回客户端）做出响应。在这种模型下，绝大部分的商业逻辑及数据操作都在服务器端执行，如输入数据的校验，用户权限的检查，数据库记录的查询、修改及删除，动态页面的生成。因此，服务器端的代码安全就显得及其重要。

由于绝大多数 Web 应用程序需要存储和查询数据，而 SQL 是数据库领域中的主流语言，因此 Web 应用中大量地使用到 SQL。SQL 注入是一种攻击方式，在这种攻击方式中，恶意代码被插入到字符串中，然后将该字符串传递到数据库服务器的实例以进行分析和执行。本文分析了 SQL 注入产生的原因及在代码中如何避免此类漏洞带来的攻击,并对不同的方案进行了比较及验证。

XSS 是另外一种经常出现在 Web 应用中的计算机安全漏洞，它允许恶意 Web 用户将代码植入到提供给其它用户使用的页面中。攻击者利用 XSS 漏洞旁路掉访问控制——例如同源策略。本文分析了 XSS 产生的原因及在代码中如何避免这种类型的攻击,并对不同的方案进行了比较及验证。

即使服务器端和客户端的代码都是安全的，无明显可利用的漏洞。但由于服务器及客户端之间的数据传输必须通过很多的交换机及路由器，攻击者仍然可以从传输的通道上获取用户账户等一系列的敏感信息。因此某些含用户敏感信息的页面使用 HTTPS 协议是必须的，而很多 WEB 应用虽然使用了 HTTPS 协议，但是仍然可以使用普通 HTTP 进行访问，本文将给出强制页面使用 HTTPS 的方法。

关键字：代码安全；SQL 注入；XSS

Abstract

With the growing popularity and development of Internet, more and more applications are changing from the traditional model to the network model. People can easily do bank balance transfer through the network, and purchase products online. And most of those internet applications are web browser based, the increasing number of those applications makes the security issues gaining more and more attentions. Web applications use the traditional client/server model, client sends request to server, sever makes the response by doing some operation (such as sending the message back to client). Using this model, most of the business logic and data processing are done in server side, such as input data validation, user permission check, database record query, update and delete the dynamic page generating. Therefore, the code security in sever side is very important.

Most of web applications need querying and storing the data, and SQL is the most popular language in database realm, so the web applications use a lot of SQL.SQL injection is one kind of attack methods, using this kind of attack, malicious codes are inserted into character string as parameter, and that string will be interpreted and executed by instance of database server. This article analyzes the cause of SQL injections, and gives a few methods that could be used in coding to avoid this kind of attack，and compare and verify those different methods.

XSS is another type of computer security vulnerability typically found in Web applications that enables attackers to inject client-side script into Web pages viewed by other users. A cross-site scripting vulnerability may be used by attackers to bypass access controls such as the same origin policy. This article analyzes the cause of XSS and gives a few methods that could be used in coding to avoid this kind of attack，and compare and verify those different methods.

Even the server side and client side code are secure, no obvious secure hole could be used. But all message packages for the interaction between client and server will go through many switches and routers in Internet, so attacker can still get the sensitive information such as user account by tapping the transferring channel. So for some pages containing sensitive information, using HTTPS protocol is necessary. Although most of web application can apply HTTPS protocol, but they still allow accessing the pages using ordinary HTTP. This article will give the methods that could apply in coding to force using HTTPS protocol.

Key Words: Code Security; SQL injection; XSS

第一章 引言

随着互联网的不断普及与发展，越来越多的应用从传统模式延伸至网络。人们可以方便的通过网络进行银行转账，产品购买。这其中大部分的应用是基于 Web 浏览器的，Web 应用程序的增多使得其安全问题变得越来越受关注。

1.1 本文研究背景

随着企业和政府越来越多的业务系统采用基于 WEB 服务方式，互联网在为用户提供方便快捷的同时，针对 WEB 业务的攻击亦在迅猛增长。

Web 攻击是目前数据窃取的主要途径。在 Websense[1]实验室对 2010 年威胁的研究报告中就指出：57% 的数据泄露攻击通过 Web 实现。Web 威胁可以在用户完全没有察觉的情况下进入企业网络，从而对公司数据资产、行业信誉和关键业务构成极大威胁。即便是一些看上去并不重要的信息片段，一旦被偷窃者汇集并归纳，其后果可能导致公司内部机构设置、战略合作伙伴关系、核心客户等重要信息被泄露。类似的安全事件有：2008 年 5 月，美国东北部零售连锁店哈纳福德（Hannaford）公司系统遭到黑客入侵，大约有 420 万名顾客的信用卡账号被泄露。2010 年 6 月，美国 AT&T 网站服务器出现严重安全漏洞近 11 万名 iPad 用户信息被发布到互联网上。给苹果和 AT&T 公司带来了严重的声誉损害。2010 年末，本田美国网站遭黑客攻击，大约 490 万名用户信息被窃取，包括用户姓名、邮件地址、车牌号等均被外泄。2011 年的 CSDN 网站高达 600 多万个明文的注册邮箱账号和密码遭到曝光和外泄，成为当年中国一次重大网络安全事故。2011 年 6 月份，花旗集团(Citigroup)确认其 Citi Account Online 服务被黑客侵入，黑客因此获取了众多客户信息。花旗称约有 1%的信用卡用户受到了此次黑客袭击的影响。

由于 Web 应用采用的是典型的客户端/服务器模型，客户端向服务器发出请求，服务器通过执行操作（如将信息发回客户端）做出响应。在这种模型下，绝大部分的商业逻辑及数据操作都在服务器端执行，如输入数据的校验，用户权限

的检查，数据库记录的查询、修改及删除，动态页面的生成。因此，服务器端的代码安全就显得及其重要。任何不恰当的代码或者程序的疏忽都可能使整个应用变得易受攻击。比较有经验的黑客就可以通过尝试一些程序员容易犯的错误来攻击站点，从而获取到一些敏感信息或者破坏站点的数据。其中程序员常犯的错误包括 SQL 注入、XSS、没有对用户权限进行控制等等。OWASP 作为开放式 Web 应用程序安全项目（OWASP，Open Web Application Security Project）组织，它提供有关计算机和互联网应用程序的公正、实际、有成本效益的信息。其主要目标是研议协助解决 Web 应用安全的标准、工具和技术文件，长期致力于协助政府或企业了解并改善网页应用程序与网页服务的安全性。在其统计出的 2010 年 10 大 Web 应用程序安全威胁中，可以看到 SQL 注入及 XSS(跨站脚本)依然高居榜首。[2]

图 1.1.1 OWASP 2010 年 10 大 Web 应用程序安全威胁

由于 Web 协议的开放性，Web 应用的服务端可以使用任何编程语言及开发环境，如 Perl，C，Java，.Net，Python，Ruby。本文所使用的是 J2EE 环境，

J2EE 是一种利用 Java 2 平台来简化企业解决方案的开发、部署和管理相关的复杂问题的体系结构。由于 J2EE 建立在 JAVA2 平台标准版（J2SE）的基础上，所以具备了 J2SE 的所有优点和功能。例如"编写一次、随处运行"的特性、方便存取数据库的 JDBC API、CORBA 技术以及能够在 Internet 应用中保护数据的安全模式等等，同时还提供了对 EJB(Enterprise JavaBeans)、Java Servlets API、

JSP（Java Server Pages）以及 XML 技术的全面支持。其最终目的就是成为一个能够使企业开发者大幅缩短投放市场时间的体系结构。

J2EE 架构下的三层结构是比较成熟及实用的模型。三层体系结构，即用户层、应用层和数据库服务器。用户层主要指用户界面，它要求尽可能的简单，使最终用户不需要进行任何培训就能方便地访问信息；第二层就是应用服务器，也就是常说的中间件，所有的应用系统、应用逻辑、控制都在这一层，系统的复杂性也主要体现在应用层；最后的数据库服务器存储大量的数据信息和数据逻辑，所有与数据有关的安全、完整性控制、数据的一致性、并发操作等都是在第三层完成。图 1.1.2 展示了一个典型的 J2EE 应用的简化架构图。

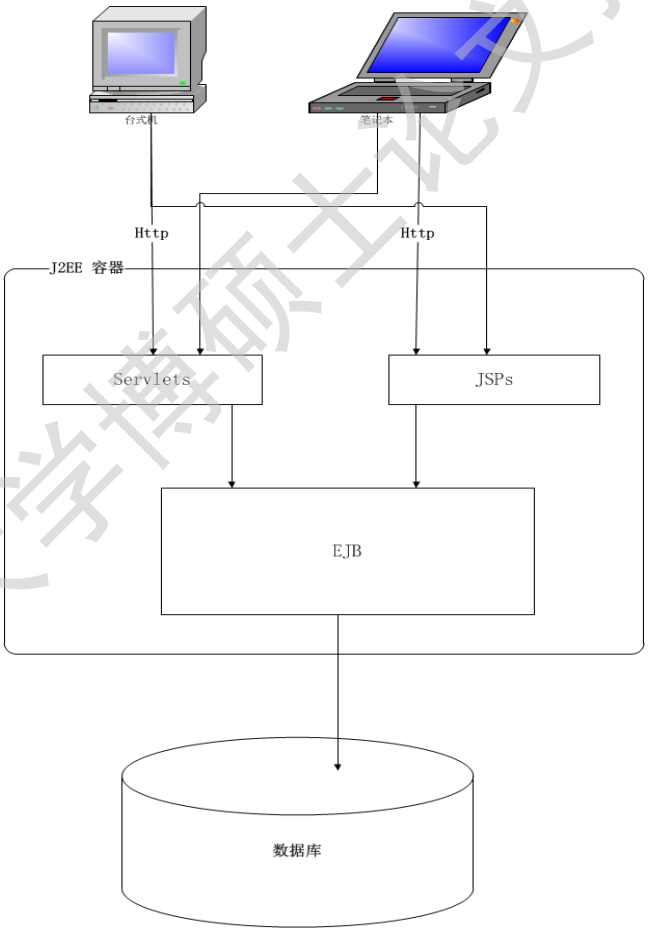

图 1.1.2 J2EE 下的三层服务模型

1.2 国内外研究现状

针对 Web 应用存在的一系列代码安全问题，国内外学术界对其进行了深入的研究，特别是对最为常见的 SQL 注入及 XSS 跨站脚本攻击。

SQL 注入攻击在国外最早出现于 1999 年，而我国在 2002 年开始出现这种攻击技术。目前还没有对 SQL 注入技术的标准定义，微软中国技术中心从两个方面进行了描述[3]：

- 脚本注入式的攻击。
- 恶意用户输入用来影响被执行的 SQL 脚本。

据《SQL 注入十年演变及防护策略》一文称，1998 年 12 月，Rain Forest Puppy(RFP)在著名安全杂志《Phrack》第 54 期上发表文章《NT Web 技术漏洞》，首先提到应用 SQL 注入的攻击技术。值得注意的是，当时 RFP 在文中并没有使用“SQL 注入”这一术语；1999 年 2 月，Allaire 发出安全通告《动态查询中的多 SQL 指令》，讨论 SQL 注入攻击这类安全威胁[4]。随着对 SQL 注入研究的不断深入，逐步总结出了一些较为常用的检测与防范措施[5]。

其中，比较常见的检测方法有：

- 数据库检查：通过查看数据库中最近新建的表的结构和内容，可以判断是否曾经发生过 SQL 注入攻击。
- Web 服务器日志检查：通过查看 Web 服务器的日志文件，判断是否发生过 SQL 注入攻击。
- 其它相关信息判断：通过查看系统管理员账户、远程终端服务器开启情况、系统最近日期产生的一些文件等信息来判断。

而比较常用的防范措施有[6,7]：

- 国内研究者提出了各种各样的防范模型，比如在客户端和服务器进行检测的 SQL 注入攻击检测/防御/备案模型[8]。
- 屏蔽出错信息。
- 对用户输入进行过滤处理。
- 使用参数化查询或存储过程。
- 目录最小化权限设置。

- 对包含敏感信息的数据进行加密后再进行存储。

XSS 攻击和 SQL 注入攻击类似，都是由于 Web 业务的代码编写人员不严谨的字符限制而导致的。当某个站点允许用户提交 Java Script 脚本（这在 Web 2.0 年代非常普遍），而又没有对这些脚本进行严格分析，就有可能存在 XSS 漏洞。1999 年，Georgi Guninski 和 David Ross 联合发表了第一篇关于 XSS 威胁标题为“脚本注入”的论文。目前针对 XSS 的安全研究有漏洞挖掘和攻击防范两个方面。

主要的 XSS 漏洞挖掘技术有[9-12]：

- 静态分析技术：包括源代码审核和二进制审核。
- Fuzzing 技术：一种自动化的软件测试技术，它使用大量的测试数据作为应用程序的输入，用于寻找应用程序的漏洞。
- 动态调试技术：在调试器环境中运行目标程序，并在运行时分析其反汇编代码，利用反汇编技术鉴别可能出现漏洞的代码。

常用的 XSS 防范措施有：

- 不信任用户提交的任何内容，对所有用户提交内容进行可靠的输入验证。
- 确认接收的内容被妥善的规范化，仅包含最小的、安全的 Tag(没有 JavaScript)，去掉任何对远程内容的引用（尤其是样式表和 JavaScript)。
- Cookie 防盗。避免直接在 Cookie 中泄露用户隐私，例如 email、密码等等。

1.3 研究内容及章节安排

针对 WEB 应用中最为常见的安全问题 SQL 注入及跨站脚本攻击，本文将在研究其产生的原因的基础上，对各种避免方法进行深入剖析，并在工程中进行实践及应用，并对应用前后进行测试，测试包含了人工构造测试案例及使用第三方工具。SQL 注入将使用 Safe3 SQL injector, 跨站脚本攻击部分将使用 XSS Me 工具。关于 HTTPS 协议，多数网站仍不能正确应用，都存在或多或少的问题，本文也将对这一方面进行剖析研究，并给出强制使用 HTTPS 协议的办法。

本文分为五章，主要内容如下：

第一章：引言。主要简单介绍 Web 应用程序的安全问题及章节安排

第二章：SQL 注入引起的代码安全问题研究。主要对 SQL 注入产生原因、攻击流程、利用方式及避免方法进行研究。

第三章：XSS 引起的代码安全问题研究。主要对 XSS 产生原因、利用方式及避免方法进行研究。

第四章：未使用 HTTPS 引起的代码安全问题研究。主要对使用 HTTPS 的必要性及怎样对一些含敏感信息页面强制使用 HTTPS 进行研究.

第五章：结束语。

第二章 SQL 注入引起的代码安全问题研究

由于绝大多数 WEB 应用程序需要存储和查询数据，而 SQL 是数据库领域中的主流语言，因此 WEB 应用中大量地使用到 SQL。代码中 SQL 使用不当造成的漏洞，能够使攻击者对其进行利用，从而获取一系列的敏感信息，照成信息的泄漏，而一些恶意的攻击者甚至能删除所有数据，让互联网应用的拥有者蒙受巨大的损失。而由于 SQL 注入的利用方式易于操作，一旦网站的服务端程序存在漏洞，攻击者并不需要太高的技巧就能实现一系列的攻击，因此如何避免 SQL 注入攻击，对于任何的互联网应用提供者来说，都是需要对其进行相应研究并且积极应用到服务器端的代码中。而不同的避免 SQL 注入方式有各自的优缺点，不同的应用可以根据实际情况选择不同的方式。

在本章中，先对 SQL 进行了简单的介绍。在 SQL 注入攻击一节中，首先将对 SQL 注入攻击做相应简介。然后分析其产生原因，攻击流程及一些较为常见的利用方式。在对 SQL 注入攻击有了相应理解之后，在 2.3 章节中将详细分析避免 SQL 注入的几种方法。在 2.4 章节中会对不同的方法进行比较分析，并采用其中一种方法进行验证。

2.1 SQL 语言

结构化查询语言(Structured Query Language）最早是 IBM 的圣约瑟研究实验室为其关系数据库管理系统 SYSTEM R 开发的一种查询语言，它的前身是 SQUARE 语言。SQL 语言结构简洁，功能强大，简单易学，所以自从 IBM 公司 1981 年推出以来，SQL 语言得到了广泛的应用。如今无论是像 Oracle、Sybase、DB2、Informix、SQL Server 这些大型的数据库管理系统，还是像 Visual Foxpro、PowerBuilder 这些 PC 上常用的数据库开发系统，都支持 SQL 语言作为查询语言。SQL 语言已经成为数据库领域中的一个主流语言。[13]

可以把 SQL 分为两个部分：数据操作语言 (DML) 和 数据定义语言 (DDL)。